I. INTRODUCTION

It is well known [1,2,3] that many existing data communication systems use a class of error control techniques known as automatic-repeat-request (ARQ) [1] for dealing with transmission errors. ARQ is often preferred over forward-error-correction (FEC) [1] because it is possible to build ARQ systems that are simple, reliable and essentially insensitive to the kinds of errors that occur on real channels. Burton and Sullivan [1] and others [2,3] have argued for the use of ARQ systems, and have pointed out the limitations of FEC schemes.

Although ARQ systems work well for low error rates, their performance as measured by the effective information rate[1] deteriorates steadily with increasing error probability. In addition, long delays due to many retransmissions are not infrequent in ARQ when dealing with burst error channels. FEC systems on the other hand usually have to sacrifice the rate considerably in order to achieve reliability comparable to ARQ. This paper proposes an error control technique which is an extension of the ARQ idea, and is capable of giving the reliability of ARQ without a corresponding decline in information rate for high error probabilities.

The basic idea behind the technique is that a received block which is determined to contain errors should not be discarded, because it contains information that could be used for error control. If the retransmission of this block is also in error, the collective information present in the first

---

[1] This is the number of new information bits transmitted divided by the total number of channel bits used. The total number of bits used does not include those wasted during idle time.

block and its retransmission could be used to correct non-overlapping errors in the two blocks. In case correction is not possible using the two blocks, additional retransmissions are needed till enough repetition redundancy is present to allow correction. An algorithm has been developed for determining the transmitted message from two or more of its corrupted copies. For this algorithm the probability of uncorrectable error decreases rapidly with the number of copies available for doing correction, and the probability of undetected error can be made negligible. Thus the technique retains the basic reliability of ARQ and may be applied in situations where it becomes unrealistic to use ARQ or FEC because of the low rates achievable.

Section II describes an ARQ with memory (MARQ) system based on this idea. The performance of such a system is examined in Section III and the results of a simulation study on a packet broadcasting channel [4] are used to compare the rate and message delay distribution of MARQ with those of ARQ.

II. SYSTEM DESCRIPTION

The ARQ with Memory (MARQ) system (Fig. 1) is basically an ARQ system which is provided with additional logic and memory at the receiver to perform error correction using retransmissions. The actual retransmission strategy is of little importance in this discussion, and for the rest of the paper, use of the feedback channel will remain implicit.

At the transmitting end, messages are blocked into k-bit words and transformed into longer n-bit codewords from a linear block code $C(n,k)$. For a given message $\bar{m}$, the transmitted codeword $\bar{v}$ is saved in the transmitter buffer, and is retransmitted as many times as indicated by the receiver. The channel adds[1] an n-bit error pattern $\bar{e}$ to every transmitted word $\bar{v}$ to give the received word $\bar{r}$. Let $\bar{r}_1,\bar{r}_2,\ldots$ denote successive received words corresponding to a given message $\bar{m}$, and let $\bar{s}_1,\bar{s}_2,\ldots$ be the corresponding syndromes. $\bar{s}_i$ is defined by $\bar{r}_i H^T$ where H is the parity check matrix of $C(n,k)$.

Decoder operation is illustrated by the state diagram (Fig. 2). States 1,2,3... represent successive iterations of the decoding algorithm, and edges represent possible transitions. Each edge is labelled by the corresponding probability of occurrence. In decoding a particular message, the decoder starts in state 1 with the first received word $\bar{r}_1$ present in its buffer and proceeds according to the following rules (i denotes the current state number).

a) If $s_i=\bar{0}$, assume $\bar{r}_i=\bar{v}$ and go to state 1. Decoding is complete.

b) If $r_i\neq\bar{0}$, save $\bar{r}_i$ in the buffer and apply repetition redundancy correction (Algorithm A1) using the buffer contents $\bar{r}_1,\bar{r}_2,\ldots\bar{r}_i$. If correction

---

[1] Unless otherwise specified, all addition of vectors is modulo-2, bit-by-bit.

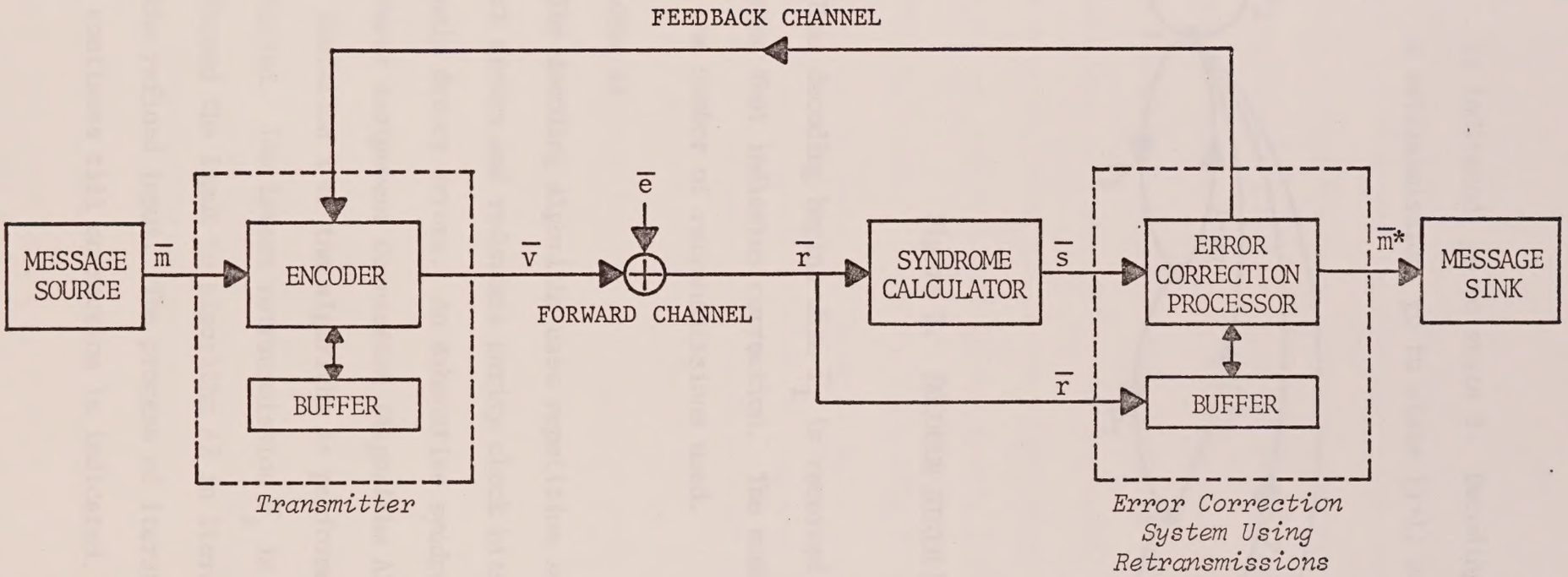

Figure 1. ARQ-WITH-MEMORY SYSTEM

is indicated, go to state 1. Decoding is complete. If not, request a retransmission, go to state (i+1) and reapply the above rules.

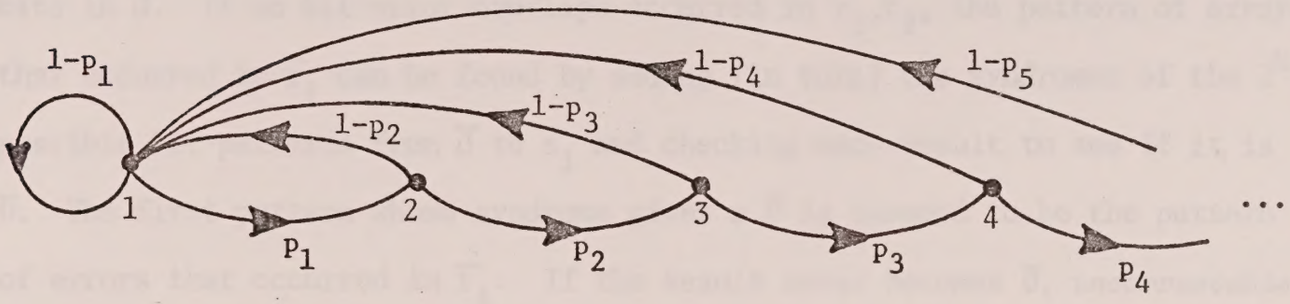

Figure 2. DECODER STATE DIAGRAM

Thus decoding begins when $\bar{r}_1$ is received and is completed in the first iteration that indicates correction. The number of this iteration minus one gives the number of retransmissions used.

Algorithm A1

The decoding algorithm uses repetition redundancy (retransmissions) to correct errors and redundant parity check bits in the received words to simultaneously detect errors. An exhaustive syndrome search technique which we term *Burst Assignment Correction* (Algorithm A2) forms the core of this algorithm. Iteration i of the algorithm is performed only if all previous iterations have failed. The latest retransmission $\bar{r}_i$ is used to refine the information that formed the input to algorithm A2 in iteration (i-1), and A2 is reapplied with the refined input. The process of iterative refinement and application of A2 continues till correction is indicated.

A) Burst-Assignment-Correction

The basic idea: Consider $\bar{r}_1, \bar{r}_2$ the first received word and its retransmission. Let $\bar{d}$ denote their modulo-2 sum, and let k be the number of nonzero bits in $\bar{d}$. If no bit-error overlaps occurred in $\bar{r}_1, \bar{r}_2$, the pattern of errors that occurred in $\bar{r}_1$ can be found by adding (in turn) the syndromes of the 2^k possible bit-patterns from $\bar{d}$ to $\bar{s}_1$ and checking each result to see if it is $\bar{0}$. The first pattern whose syndrome gives a $\bar{0}$ is assumed to be the pattern of errors that occurred in $\bar{r}_1$. If the result never becomes $\bar{0}$, *uncorrectable* errors have occurred, and if the result becomes $\bar{0}$ but the pattern found is not the right one, *undetectable* errors have occurred.

When k is large, a search from among 2^k patterns is impractical. A reduction in the number of patterns to be searched (at the expense of correction ability) can be obtained by breaking $\bar{d}$ into bursts[1], $\bar{b}_1, \bar{b}_2, \ldots \bar{b}_\ell$ with guard band g, and making the search on syndromes of the 2^ℓ patterns possible with *bursts* instead of bits. In general, increasing g decreases ℓ but increases the probability of uncorrectable error. Thus g can be used to control the computation complexity to some extent. Here uncorrectable errors have occurred whenever a burst from $\bar{d}$ has errors from both $\bar{r}_1$ and $\bar{r}_2$.

The Algorithm:

Input: $\bar{s}_1$, the syndrome of $\bar{r}_1$; a list $B=\{\bar{b}_1, \bar{b}_2 \ldots \bar{b}_\ell\}$ of bursts and a list S of syndromes of the bursts in B.

Output: $\bar{e}$, the algorithm's estimate of the error pattern in $\bar{r}_1$, or an indication of uncorrectable errors.

---

[1] A burst with guard band g is defined as a maximal length string of 1's and 0's beginning and ending with a 1, not containing g or more consecutive 0's.

Step 1: Let $\{\bar{b}'_1,\bar{b}'_2,\ldots\bar{b}'_j\}$ $j\leq\ell$ be a subset of B such that the sum of the corresponding syndromes from S when added to $\bar{s}_1$ gives $\bar{0}$. If no such subset exists, output "uncorrectable errors" and stop.

Step 2: Output $\bar{e}=\bar{b}'_1+\bar{b}'_2+\ldots+\bar{b}'_j$ as the algorithms estimate of the error pattern in $\bar{r}_1$ and stop.

B) The Refinement Process

If algorithm A2 fails in state i-1, the latest retransmission $\bar{r}_i$ is used in state i to update B and S, the list of bursts and corresponding syndromes. Let $\bar{d}'=\bar{r}_1+\bar{r}_i$. For each burst $\bar{b}$ in $\bar{d}$, if $\bar{b}$ overlaps with some $\bar{b}'$ in $\bar{d}'$, then $\bar{b}+\bar{b}'$ is appended to B and the syndrome of $\bar{b}+\bar{b}'$ is appended to S. The motivation behind doing this is that after refinement, algorithm A2 will fail to correct the errors if there exists at least one set of i bursts, one each from $\bar{r}_1,\bar{r}_2\ldots\bar{r}_i$, such that every burst overlaps with every other. We call the existence of one such set a *degree-i overlap*. Heuristically, it can be seen that if the positions in which errors occur are independent from block to block, the probability of a degree-i overlap is a rapidly decreasing function of i, and the probability of uncorrectable errors therefore becomes small very quickly.

C) An Extension:

Consider now a more general MARQ system which has a simple built-in FEC capability inserted *before* algorithm A1. Let C_c denote the set of n-bit vectors correctable using this capability. No forward error correction is now simply represented by the special case $C_c=C$. We next analyze the general MARQ system.

III. PERFORMANCE OF MARQ

As for ARQ, the key parameter in evaluating an MARQ system is the effective information rate R. By our definition,

$$R = \lim_{T\to\infty} \frac{\text{Number of new message bits transmitted in time } (0,T)}{\text{Number of channel bits used in } (0,T) \textit{ excluding } \text{idle bits}}$$

$$= \frac{k}{n} \cdot \frac{1}{\bar{N}}$$

where $\bar{N}$ is the average number of blocks needed to convey a given message. Another parameter of interest is the probability of undetected error P_u, which gives an indication of overall system reliability.

A) Characterization of Effective Information Rate

From the state diagram of Fig. 2, the average number of blocks transmitted per message can be shown to be

$$\bar{N} = \sum_{j=1}^{\infty} j(1-p_j) \prod_{i=1}^{j-1} p_i \qquad (1)$$

Once the decoding algorithm is specified, the p_i are determined completely by channel error-statistics.

It is clear that p_i ($i \geq 1$) represents the probability that the correction algorithm will *indicate* failure in the j^{th} iteration. p_1 is simply $\Pr\{\bar{r}_1 \notin C_c\}$. For $i \geq 2$, it can be shown that $p_i < \Pr\{\text{at least one degree i overlap} \mid \text{state i}, \bar{r}_i \notin C_c\}$. In general, the number of degree i overlaps is at most equal to the number of degree (i-1) overlaps. It follows that p_i is monotonic non-decreasing with i, and $\prod_{i=1}^{j-1} p_i$, the probability that the algorithm reaches the j^{th} iteration, is at worst geometric with j.

A simple upper bound on R can be obtained by noting that the best the algorithm can perform is when $p_2=0$. Setting $p_2=0$ in (1) we get

$$\bar{N} \geq 1 + p_1$$

$$R_{MARQ} \leq \frac{k}{n} \cdot \frac{1}{1+p_1} \tag{2}$$

A lower bound on R is given by the rate achieved by hybrid ARQ (HARQ), which corresponds to MARQ without the capability of using saved blocks for doing correction.

$$R_{MARQ} \geq \frac{k}{n}\left(\frac{1}{1+p_1+p_1^2+\ldots}\right) = \frac{k}{n}(1-p_1) = R_{MARQ} \tag{3}$$

B) Characterization of Undetected Errors

Undetected errors in MARQ can arise in two distinct ways: 1) An undetected error occurs in FEC decoding (this will happen if some $\bar{r}_i \varepsilon C_c$ and $\bar{r}_i \neq \bar{v}$, the transmitted codeword); 2) An undetected error occurs in algorithm A2.

Let r=n-k be the number of parity check bits in a codeword, and m be the total number of FEC-correctable error patterns. If the code C and the FEC capability m are chosen so that all likely error patterns and most of the remaining patterns are detected, the probabilities of undetected FEC and BAC error can be conservatively estimated [5] by

$$P_{FEC} \simeq \frac{1}{2^{(r-\log_2 m)}} \quad , \qquad P_{BAC} \simeq \frac{1}{2^{(r-\ell)}}$$

where ℓ is the maximum[1] number of bursts allowed in the BAC algorithm. The

---

[1] ℓ is fixed by the maximum amount of computation that the BAC algorithm is permitted to do in one iteration.

overall probability of undetected error P_u is given by $P_{FEC}+(1-P_{FEC})P_{BAC}$. This can clearly be made negligible by choosing r large enough. The point, however, is that if $\log_2 m$ and ℓ are reasonable, P_u can be made negligible without sacrificing the fixed component (k/n) of the rate R.

C) Performance on a Real Channel

A simulation of the MARQ system was performed using error data from the ATS-1 VHF satellite channel being operated in the ALOHA random-access burst mode [6]. The channel is characterized by a mix of random and burst errors, and is one instance where MARQ gives a considerable improvement over other schemes. The block length used was 640 bits and an FEC capability of double adjacent and single errors was incorporated.

Figure 3 shows the upper and lower bounds on R_{MARQ} plotted against p_1, the probability of uncorrectable (FEC) errors, and actual data points obtained from the simulation. Each data point corresponds to a simulation run on a set of approximately 1000 error patterns, and a different value of p_1. It is interesting to note that all of the points are clustered around the upper bound on R_{MARQ}, indicating that the system is performing near optimum on this type of channel. Some of the points are actually above the upper bound. The reason for this is, of course, that the data base for each point is not large enough. The greatest improvement in R over HARQ is obtained for large p_1, since for small p_1 algorithm A1 is rarely invoked, and thus does not significantly affect R.

The improvement in distribution of number of retransmission per message is even more interesting. Figure 4 shows a plot of the distributions obtained for MARQ and HARQ for the same error data. On a sample size of $\approx 10^4$ messages,

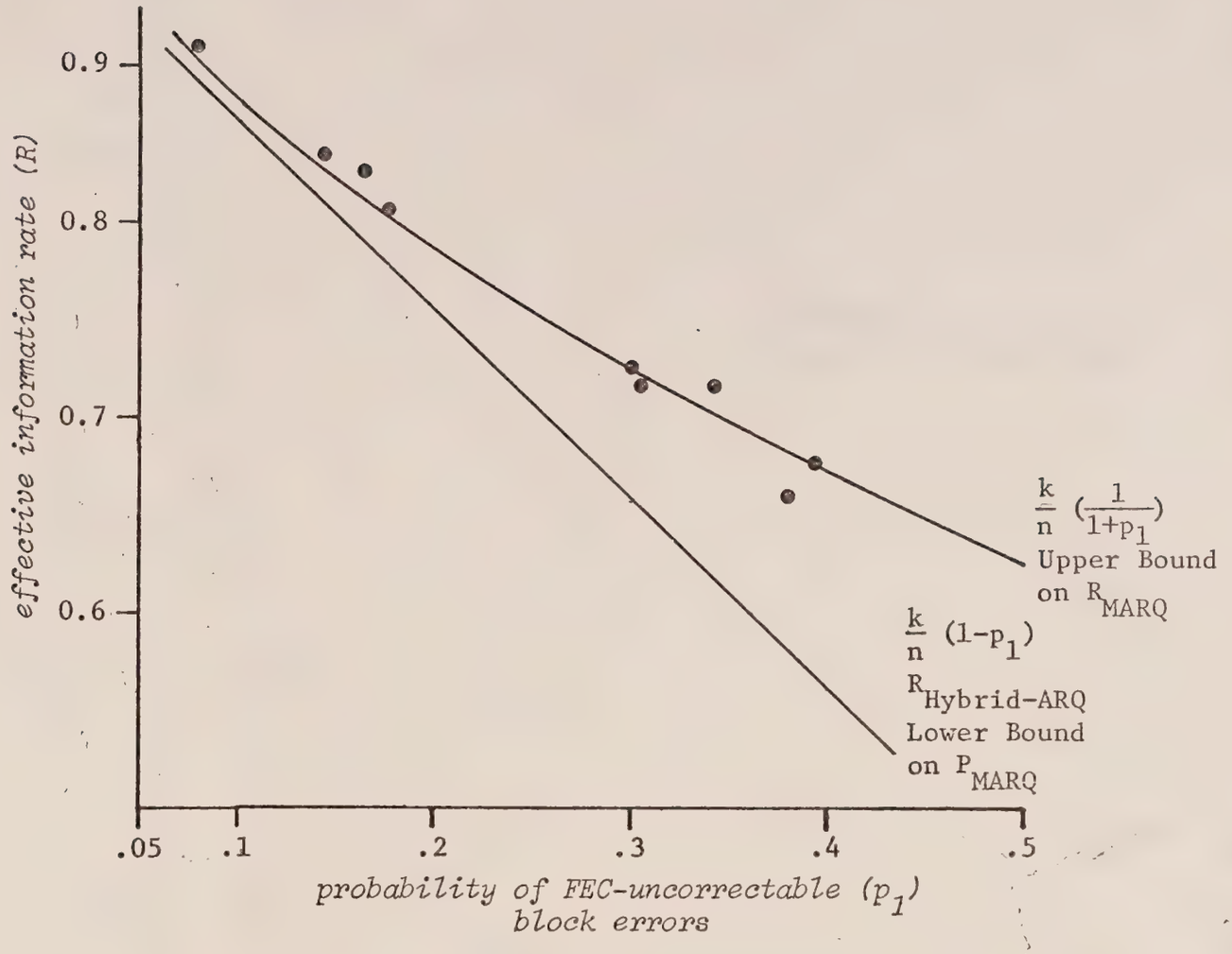

Figure 3

COMPARISON OF EXPERIMENTALLY OBTAINED DATA POINTS WITH UPPER AND LOWER BOUNDS ON R_{MARQ}

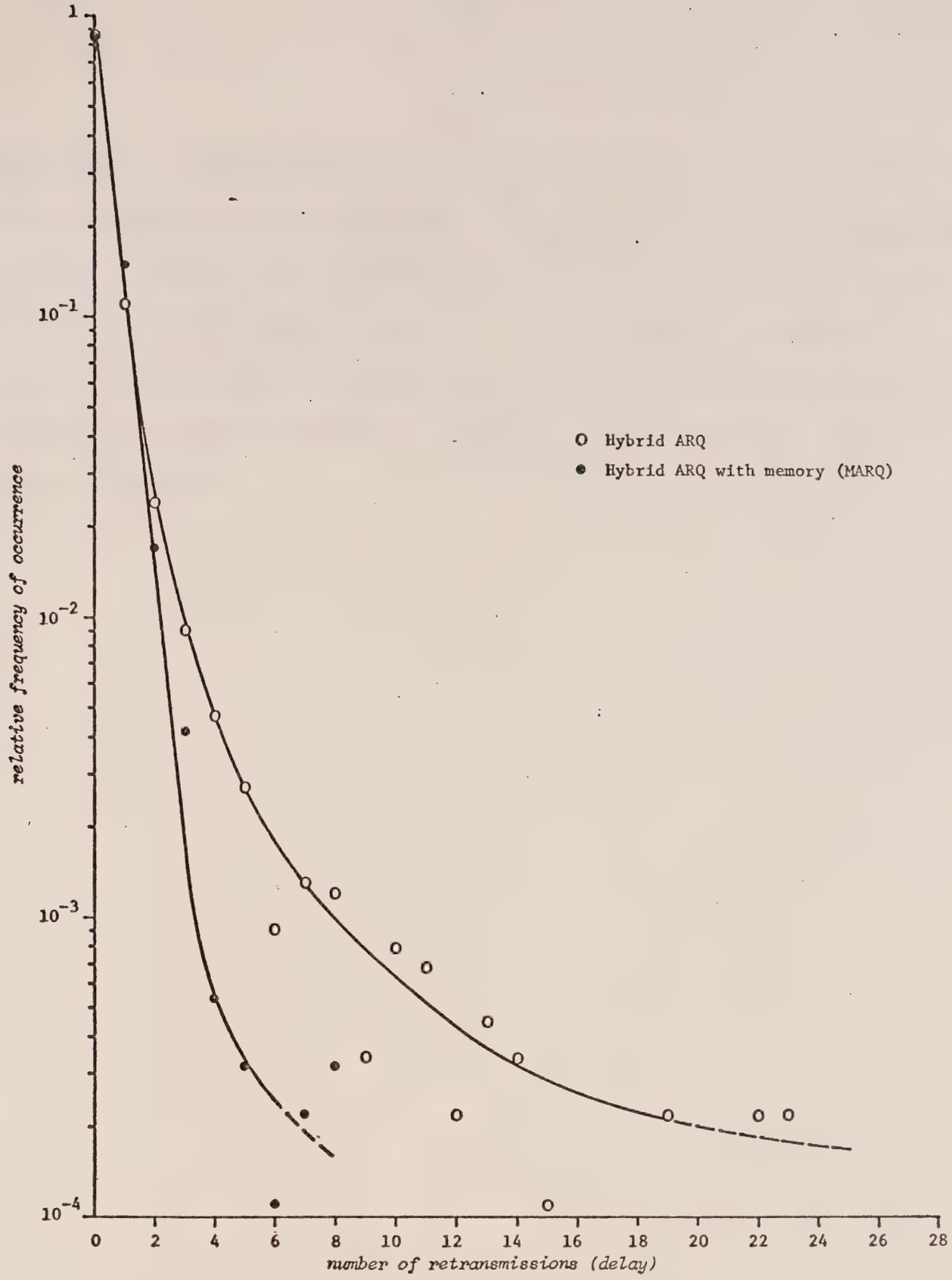

Figure 4

COMPARISON OF DELAY DISTRIBUTION FOR MARQ AND HYBRID-ARQ

more than three retransmissions were recorded only about ten times, and more than eight retransmissions never--an improvement of an order of magnitude over HARQ for small number of retransmissions, and larger for more retransmissions. The fact that the delay distribution changes so radically is especially significant for systems using ARQ where long message delay is expensive or becomes intolerable. This is true for example of a small-terminal/computer interface.

IV. CONCLUSION

The main attempt in this paper has been to show how retransmission redundancy can be used for error control, and demonstrate that the technique is basically an improvement on memoryless-ARQ. Although the correction algorithm is suited to channels with dependent errors, and operates close to its optimum here, it can also be used for independent errors. The improvement in rate is seen to be the largest when the block probability of error is high. The result is interesting because this is precisely the region where FEC performs better than memoryless-ARQ [3], both for dependent and independent errors, although the probability of undetected error is usually much higher for FEC. If we consider the same system reliability, it is clear that one would have to go to even higher channel error probabilities for FEC to have an advantage over memoryless-ARQ. For the same reliability, then, there is a broad range of channel error probabilities where MARQ is better than FEC as well. It should, however, be noted that MARQ inherits most of the other disadvantages of ARQ, and therefore cannot be used where ARQ is limited by factors other than rate.

The method of correction can be incorporated in an existing ARQ system with little added complexity and almost no changes in protocol. Whether implementing MARQ in a given situation is worth the increase in throughput rate is, of course, technology dependent but it is clear that with the ever decreasing cost of memory and the availability of cheap computing elements, MARQ is a viable error control technique.

REFERENCES

[1] H. O. Burton, and D. D. Sullivan, "Errors and Error Control," *Proceedings of the IEEE*, Vol. 60, No. 11, November 1972.

[2] R. J. Benice and A. H. Frey, Jr., "An Analysis of Retransmission Systems," *IEEE Trans. Commun. Technol.*, Vol. COM-12, December 1964.

[3] R. J. Benice and A. H. Frey, Jr., "Comparisons of Error Control Techniques," *IEEE Trans. Commun. Technol.*, Vol. COM-12, December 1964.

[4] N. Abramson, "THE ALOHA SYSTEM" in *Computer Communication Networks*, Prentice-Hall, 1973, pp. 501-518.

[5] P. Sindhu, "An Improvement on ARQ Error Control," *ALOHA System Technical Report* (in preparation).

[6] D. Wax, "Status Report on UH/ALOHA Participation in the ATS-1 Computer Communication Experiment," *ALOHA System Technical Report B74-8*, September 1974.

[7] S. Lin, *An Introduction to Error Correcting Codes*, Prentice-Hall, 1970.

[8] W. W. Peterson, and E. J. Weldon, Jr., *Error Correcting Codes*, 2nd edition, The MIT Press, Cambridge, Massachusetts, 1970.